函数名不区分大小写，这点与变量名不同。

如果希望在函数内部使用函数外面的变量，需要在函数内部使用关键字global修饰变量。

$var=100;

function test() {

global $var; //在test()范围内，使用函数外面的变量

echo "在函数内部var的值为：".$var;

}

test();

预定义变量**$GLOBALS**：它是一个包含了全部变量的全局组合数组，变量的名称就是数组的键，可以读取全局变量。

$var=100;

function test()

{

//使用$GLOBALS访问全局变量

echo "在函数内部var的值为：".$GLOBALS["var"];

}

test();

function getsum($n)

{

if($n==1) //满足条件，递归结束

return 1;

$temp=getsum($n-1); //调用自身，使n的值递减

return $temp+$n;

}

echo "sum=".getsum(4);

注：getsum()函数被调用了4次，并且每次调用时，n的值都会递减。当n的值为1时，所有递归调用的函数都会以相反的顺序相继结束，所有的返回值会进行累加。

**Explode()函数：分隔字符串**

$str="apple,pear,banana,orange";

$arr=explode(",",$str); //用逗号对str字符串进行分隔

echo "第一次分隔的结果为：";

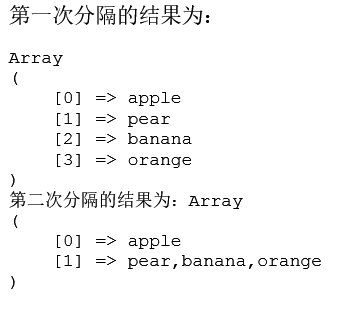
echo "<pre>";

print\_r($arr)."<br>";

echo "第二次分隔的结果为：";

$arr=explode(",",$str,2); //用逗号对str字符串进行分隔，限制返回字符串个数为2

print\_r($arr);



implode()函数：将字符数组拼接成一个新的字符串。

$color\_arr=array("red","green","blue");

$str="accelworld";

$color\_str=implode($str,$color\_arr);

echo "color\_str=".$color\_str;

返回的值：color\_str=redaccelworldgreenaccelworldblue

**strcmp()函数**：判断两个字符串的大小

例子：

$str1="abcde";

$str2="abcDe";

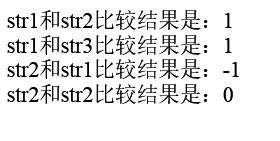
$str3="abcd";

echo "str1和str2比较结果是：".strcmp($str1,$str2)."<br>";

echo "str1和str3比较结果是：".strcmp($str1,$str3)."<br>"; //长度大者大

echo "str2和str1比较结果是：".strcmp($str2,$str1)."<br>";

echo "str2和str2比较结果是：".strcmp($str2,$str2)."<br>"; //可以比较相同字符串



还需要再加一个case循环，说明各数字的含义，不然客户看不懂

**str\_replace()函数**：对字符进行替换操作

例子：

$str1="i like play football, and he is also like play football";

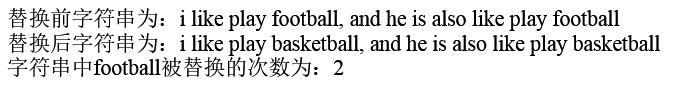
$str2="basketball";

echo "替换前字符串为：".$str1."<br>";

$str=str\_replace("football",$str2,$str1,$count);

echo "替换后字符串为：".$str."<br>";

echo "字符串中football被替换的次数为：".$count."<br>";

**substr()函数**：截取字符串。

例子：

$str="this is a string";

$str1=substr($str,0,4);

$str2=substr($str,0);

$str3=substr($str,0,-4);

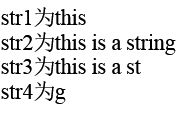
$str4=substr($str,-1);

echo "str1为".$str1."<br>"; //从第一个字符开始，截取4个字符

echo "str2为".$str2."<br>"; //从第一个字符开始截取，直到字符串的末尾

echo "str3为".$str3."<br>"; //从第一个字符开始截取，直到字符串末端第4个字符

echo "str4为".$str4."<br>"; //从字符串末端返回1个字符



**strlen()函数**：获取字符串的长度

声明：int strlen( string $str)

参数$str用于表示待获取字符串的长度

$str1="abcd";

$str2="中文字符串";

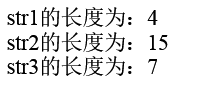
$str3="空 格";

echo "str1的长度为：".strlen($str1)."<br>";

echo "str2的长度为：".strlen($str2)."<br>";

echo "str3的长度为：".strlen($str3)."<br>";

注意：单个英文字符和一个空格的长度均为1，一个中文字符的长度为3。



**trim()函数**：用于去除字符串中首尾两端的空白字符。

$str=" hello world ";

echo "未调用trim函数：".$str;

echo "<br>";

echo "调用trim函数：".trim($str);

**格式化输出**

date()函数：格式化日期时间

声明：string date( string $format [,int $timestamp])

返回将整数$timestamp 按照给定的格式字串而产生的字符串。如果没有给出时间戳则使用本地当前时间。换句话说，$timestamp 是可选的，默认值为 time() 。

例子：echo date("Y年m月d日H时i分s秒",time())."<br>";

echo "Today is ".date("l",time())."<br>";

echo date("l dS \of F Y h:i:s A"); //非format字符要加上反斜线来转义，否则会按照format字符进行解释

mktime()函数：返回一个指定的时间戳。

声明：int mktime ([int $hour [,int $minute [,int $second [,int $month [,int $day [,int $year] ] ] ] ] ])

参数可以从右向左省略，任何省略的参数会被设置成本地日期和时间的当前值。

echo date("Y-m-d H:i:s",mktime())."<br>";

echo date("Y-m-d H:i:s",mktime(14))."<br>";

echo date("Y-m-d H:i:s",mktime(14,28))."<br>";

echo date("Y-m-d H:i:s",mktime(14,28,56))."<br>";

echo date("Y-m-d H:i:s",mktime(14,28,56,4))."<br>";

echo date("Y-m-d H:i:s",mktime(14,28,56,4,10))."<br>";

echo date("Y-m-d H:i:s",mktime(14,28,56,4,10,2013))."<br>";

Y-m-d H:i:s 代表年，月，日 时，分，秒

注意大小写，新手请按部就班的学习。

由于时间戳是一个整数如1365604136，以32位二进制数来表示，所以使用date()函数用于格式化输出。

**获取系统当前时间**

time()函数

声明：int time( void )

该函数无参数，php默认时区为美国时区UTC，根据需要更改时区即可。

echo date("Y-m-d H:i:s",time())."<br>";

**获取用户提交的时间**

strtotime()函数：用于将字符串转化成时间戳。

echo date("Y-m-d H:i:s",strtotime("2016-10-16 17:45:12"))."<br>";

echo date("Y-m-d H:i:s",strtotime("+1 day"))."<br>";

echo date("Y-m-d H:i:s",strtotime("next monday"))."<br>";

先不管那么多，就照上面的用法用着先。。。

**获取精确时间**

microtime()：与time相似，用于处理更精确的时间。

声明：mixed microtime ( [ bool $get\_as\_float ] )

该函数当前返回UNIX时间戳以及微秒数，参数$get\_as\_float是可选参数，设置为true将返回一个浮点数，省略则以msec sec格式返回一个字符串，其中mesc是微秒部分，sec是秒数，但都是以秒为单位返回的。

function microtime\_float()

{

list($usec,$sec)=explode(" ",microtime());

return ((float)$usec+(float)$sec);

}

$time\_start=microtime\_float();

usleep(1000);

$time\_end=microtime\_float();

$time=$time\_end-$time\_start;

echo "执行该脚本花费了($time)秒";

usleep ：以指定的微秒数延缓程序的执行。

Explode分割函数分割microtime返回的字符串，再分别记录下延迟前后的时间戳，算出两者之间的时间差值